**IJCCS**, Vol.7, No.2, July 2013, pp. 167~176
ISSN: 1978-1520 

# Penyusunan Notasi Musik dengan Menggunakan Onset Detection pada Sinyal Audio

**Anindita Suryarasmi*[1], Reza Pulungan[2]**
[1]Program Studi Komputer dan Sistem Informasi, Sekolah Vokasi, UGM, Yogyakarta
[2]Jurusan Ilmu Komputer dan Elektronika, FMIPA UGM, Yogyakarta
e-mail: ***[1]aninditasuryarasmi@gmail.com**, [2]pulungan@ugm.ac.id

***Abstrak***

*Notasi musik merupakan dokumentasi tertulis dari sebuah lagu. Walaupun notasi musik telah umum digunakan, namun tidak semua orang yang berkecimpung di dalam dunia musik memahami bagaimana notasi musik dituliskan. Penelitian ini menawarkan penyusunan notasi music secara otomatis dengan mengimplementasikan metode onset detection.*

*Hal mendasar yang harus diketahui dalam pembuatan notasi musik adalah durasi serta nada yang dimainkan. Dengan menggunakan mendeteksi onset dari data audio, jarak antar pukulan dapat diketahui. Dengan demikian maka durasi permainan pun bisa dihitung. Hasil dari pencarian durasi tersebut diolah kembali untuk menciptakan objek-objek note yang disusun dalam notasi musik. Sistem menghasilkan keluaran berupa file dengan format musicXML. Dengan format ini maka hasil keluaran sistem akan bersifat dinamis dan dapat diolah kembali dengan music editor yang mendukung format file tersebut.*

*Hasil penelitian menunjukkan akurasi yang tinggi dalam pengenalan pola permainan yang berhubungan dengan durasi setiap note hingga mencapai 99.62%.*

***Kata kunci***— *notasi musik, onset detection, musicXML*

***Abstract***

*Musical notation is written documentation of a music. Even though musical notation is commonly used, not every musician knows how to write a musical notation. This work offers automatic musical notation generation from audio signal using onset detection.*

*Duration and pitch of the notes are two basic parameters that have to be known in order to generate music notation. This work implemented onset detection method to recognize the pattern by measuring the interval between two notes. Using the interval, the duration of each notes can be calculated and used to create note objects in order to arrange a musical notation. The output of the system is a musicXML formatted file. This format allowed the output to be edited using software for music editor.*

*The result of this work shows high accuracy up to 99.62% for detecting each notes and measuring the duration.*

***Keywords***— *musical notation, onset detection, musicXML*

## 1. PENDAHULUAN

Notasi musik merupakan metode dokumentasi tertulis dari sebuah lagu yang menyimpan semua informasi mengenai bagaimana musik dimainkan. Pendokumentasian seperti ini memungkinkan seorang pemain musik dapat memainkan lagu sesuai dengan bagaimana lagu tersebut diciptakan [1]. Notasi musik dapat ditulis di atas kertas maupun menggunakan

*Received June 1st,2013; Revised July 1st, 2013; Accepted July 15th, 2013*

perangkat lunak *music editor*. Penulisan notasi musik di atas kertas memiliki kelemahan yaitu sulitnya memanipulasi notasi yang sudah ditulis, sedangkan dengan menggunakan *music editor*, notasi yang telah ditulis dapat dimanipulasi sesuai dengan kebutuhan. Setiap *music edito*r memiliki format *file* keluaran yang berbeda-beda. Perbedaan format *file* keluaran tersebut tidak memungkinan suatu notasi yang ditulis dalam satu *music editor* dibuka pada *music editor* lainnya. Untuk mengatasi hal tersebut, terdapat format *file* standar untuk penulisan notasi musik yaitu musicXML [2].

Walaupun notasi musik telah umum digunakan, namun tidak semua orang yang berkecimpung di dalam dunia musik memahami bagaimana notasi musik dituliskan. Hal tersebut dikarenakan penulisan notasi musik membutuhkan pengetahuan tersendiri [3]. Sekalipun *music editor* telah beredar luas di pasaran dan relatif mudah digunakan, namun penulisan notasi dari sebuah musik pada *music editor* tetap harus dilakukan dengan menuliskan *note* satu per satu, seperti halnya sebuah kalimat yang akan dituliskan dalam *text editor* tetap harus tetap diketik per huruf.

Sebagai alternatif penyelesaian terhadap kesulitan penulisan notasi musik, musik dapat disimpan dengan cara direkam. Dengan adanya rekaman tersebut, suatu musik dapat dimainkan berulang-ulang bahkan ditiru permainannya. Namun allternatif tersebut masih memiliki kekurangan yaitu pemain musik harus mencari sendiri informasi mengenai musik yang akan dimainkan seperti ketinggian nada serta panjangnya durasi setiap *note* dalam rekaman permainan.

Ketinggian nada beserta durasi *note* merupakan hal mendasar yang perlu diketahui dalam pembuatan notasi musik [4]. Dengan menggunakan kemampuan komputer untuk mendeteksi awal dari suatu *note* (*onset detection*), durasi setiap *note* yang dimainkan bisa dihitung. Dengan demikian, pembuatan notasi musik secara otomatis pun mungkin untuk dilakukan. Berdasar latar belakang permasalahan tersebut dilakukan penelitian mengenai bagaimana mengembangkan sebuah aplikasi yang dapat menganalisa data audio yaitu mendeteksi *onset* untuk mendapatkan durasi setiap *note* dari suatu lagu yang telah direkam, kemudian memanfaatkan data tersebut untuk penyusunan notasi musik yang dibuat dalam format musicXML agar dapat dibuka di berbagai *music editor*. Dengan adanya penelitian ini maka akan diperoleh aplikasi berbasis *desktop* untuk mengolah masukan data audio berupa rekaman suara permainan musik dan memberikan keluaran berupa notasi musik dengan format musicXML.

Berbagai penelitian telah dilakukan sebelumnya mengenai penggunaan *onset detection*, antara lain pendeteksian keberadaan suara berdasar amplitudo tertentu [5], pendeteksian *onset* dengan segmentasi berbasis *pitch* [6], pendeteksian *onset* dengan metode *peak picking* [7,8,9], penggunaan konsep *slow onset* [10], hingga penggunaan Short Time Fourier Transform dalam pendeteksian *onset* [11].

## 2. METODE PENELITIAN

### *2.1 Analisa permasalahan*

Penulisan notasi musik secara manual biasanya mengandalkan daya ingat pemain musik dalam menuliskan setiap detail nada dan pola yang dimainkan. Tidak menutup kemungkinan musik yang akan ditulis dalam notasi direkam terlebih dahulu agar dapat dimainkan berulang-ulang untuk memastikan setiap bagian dari permainan musik telah termuat dengan benar pada notasi musik.

Notasi musik dapat dituliskan secara manual di atas kertas maupun menggunakan perangkat lunak khusus yang sering disebut dengan *music editor*. *Music editor* merupakan perangkat lunak yang berfungsi untuk membantu pemain musik, komposer, maupun *arranger* dalam menuliskan maupun membaca notasi musik. Dengan menggunakan *music editor*, satu per satu *note* dimasukkan ke dalam aplikasi dengan diberikan pengaturan nada dan durasi sesuai dengan lagu yang akan didokumentasikan. Setelah peletakan *note* selesai dilakukan, notasi

musik dapat langsung dimainkan untuk mengecek kebenaran notasi yang telah dimainkan atau langsung disimpan sebagai sebuah *file* dalam komputer.

Setiap *music editor* memiliki format *file* keluaran yang berbeda-beda. Sebagai contoh, Sibelius memiliki format *file* keluaran .sib, Encore memiliki format *file* keluaran .enc, Finale memiliki format *file* keluaran .mus, MuseScore memiliki format *file* keluaran .mscz, dan sebagainya. Perbedaan format *file* keluaran tersebut tidak memungkinan suatu notasi yang ditulis dalam satu *music editor* dibuka pada *music editor* lainnya.

Sekalipun keberadaan *music editor* sangat membantu dalam penulisan notasi musik, namun tetap saja proses pengenalan nada maupun pola harus dilakukan sendiri oleh penulis. Seperti halnya menulis surat dengan menggunakan perangkat lunak pengolah kata, pengguna perangkat lunak harus memikirkan sendiri konten yang akan dituliskan, dan perangkat lunak hanya menuliskan konten tersebut sesuai dengan format yang ada untuk mempermudah dibaca ulang, maupun diproses lebih lanjut seperti dicetak, dan lain sebagainya.

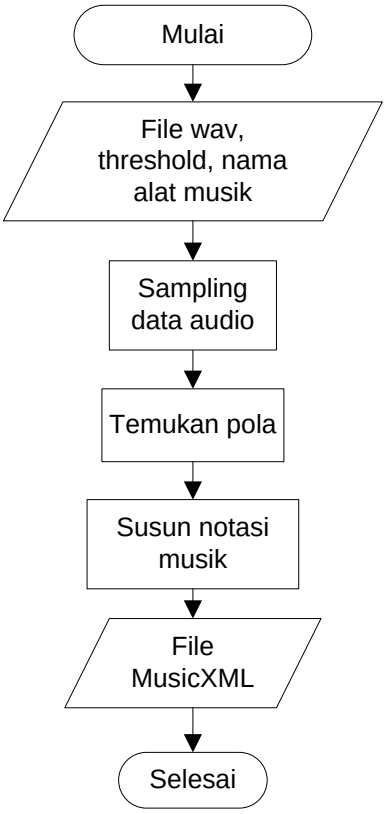
Selain penggunaan notasi musik, merekam sebuah lagu bisa menjadi alternatif lain dalam pendokumentasian lagu. Hal ini dikarenakan proses merekam jauh lebih mudah daripada proses menulis notasi. Selain itu, perangkat keras untuk merekam suatu lagu bisa bermacam-macam. Dari alat khusus perekam suara, hingga perangkat genggam yang sering dibawa kemana-mana seperti *handphone*, mp3 *player*, dan sebagainya. Maka dari itu tidak jarang pemain musik merekam permainan musik mereka sebagai dokumentasi lagu yang telah dimainkan.

*2.2 Arsitektur*

Untuk dapat membangun sistem yang mampu menganalisa file berupa rekaman suara dan menghasilkan notasi music yang dapat dibuka dengan menggunakan music editor diperlukan beberapa proses antara lain *sampling*, pencarian pola permainan, penyusunan notasi musik, dan konversi ke dalam format musicXML. Proses-proses tersebut dapat digambarkan dalam *flowchart* pada Gambar 1.

Proses *sampling* merupakan proses perubahan sinyal kontinyu menjadi sinyal diskrit. Hasil proses *sampling* akan disimpan dalam list dan digunakan untuk proses pencarian pola permainan. Di dalam proses pencarian pola permainan ini diimplementasikan *onset detection*, yaitu pencarian titik-titik waktu yang merepresentasikan sebuah pukulan terhadap alat musik. Jika titik-titik tersebut telah ditemukan, maka jarak antar titik dihitung dan diperoleh durasi dari setiap pukulan.

Data durasi yang diperoleh dari proses pencarian pola ini selanjutnya dibawa ke proses pembuatan notasi. Proses pembuatan notasi sendiri dibagi menjadi 3 bagian, yatiu pembuatan *note*, penyusunan *note*, dan konversi notasi ke musicXML. Proses pembuatan *note* merupakan proses penciptaan objek-objek *note* berdasarkan data durasi dari proses sebelumnya. *Note-note* yang diciptakan dalam proses ini kemudian disusun dalam birama-birama sesuai dengan nilai durasinya sesuai dengan aturan penyusunan ntoasi musik. Notasi musik yang berisi birama-birama yang sebelumnya telah diisi oleh notasi musik kemudian dikonversi sehingga menghasilkan *file* musicXML yang dapat dibuka di berbagai *music editor* yang mendukung format *file* ini.



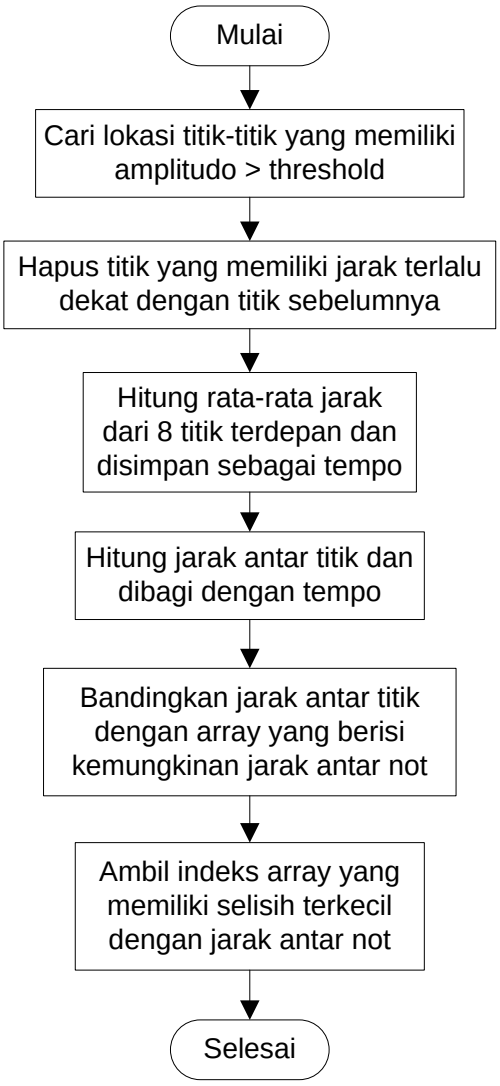

Gambar 1 Flowchart system

*2.3 Metode penyelesaian masalah*

Dalam implementasi sistem, proses pertama yang harus dilakukan adalah sampling. Proses sampling dilakukan dengan memanggil prosedur read() milik objek AudioInputStream. Hasil dari proses sampling ini adalah array yang berisi data audio yang bersifat diskrit. Data ini masih harus diolah kembali dengan menyejajarkan dua buah data integer 8 bit menjadi menjadi 1 buah data integer 16 bit. Hal ini dilakukan karena hasil pemanggilan prosedur read menghasilkan data integer 8 bit, padahal data yang sebenarnya adalah data integer 16 bit.

Hasil dari proses sampling kemudian diolah kembali untuk dapat dicari titik-titik yang merepresentasikan sebuah pukulan terhadap alat musik. Titik-titik tersebut ditemukan dengan cara menelusuri satu per satu data yang berada di dalam *list* hasil sampling untuk memperoleh data yang nilainya melebihi nilai *threshold* yang dimasukkan oleh pengguna. Titik-titik yang nilainya melebihi *threshold* ini kemudian disimpan dalam sebuah *list*. Setelah semua titik ditemukan, selanjutnya list tersebut harus diseleksi ulang karena sangat memungkinkan adanya 2 titik yang berdekatan yang sebenarnya berasal dari 1 suara. Oleh karena itu, jika 2 buah titik memiliki jarak yang terlalu dekat, maka titik kedua akan dihapus karena dianggap masih 1 suara dengan titik yang sebelumnya.

Proses yang harus dilakukan selanjutnya adalah pencarian tempo. Tempo merupakan kecepatan permainan musik. Dalam penelitian ini, data tempo disimpan sebagai panjang waktu selama 1 ketuk. Sebelum pemain music memainkan permainan music, terlebih dahulu pemain music memainkan 8 ketuk pukulan in tempo. Delapan ketuk pertama ini akan dihitung jarak rata-ratanya dan disimpan sebagai tempo. Tempo yang telah dihitung kemudian disimpan dan digunakan dalam proses selanjutnya.



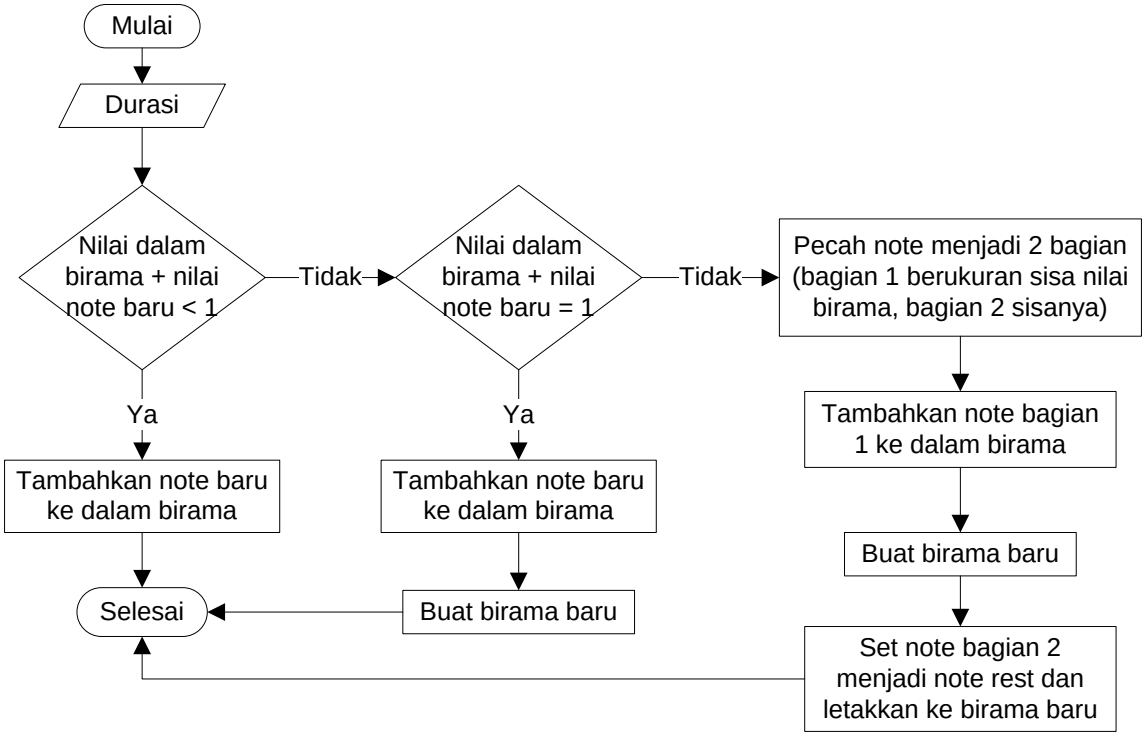

Gambar 2 *Flowchart* proses pencarian pola permainan

List berisi lokasi titik-titik hasil pencarian sebelumnya kemudian diproses untuk menghasilkan jarak antar titik. Jarak antar titik tersebut kemudian dibandingkan dengan jarak pada tempo. Jika jarak hasil perhitungan nilainya sama dengan tempo, maka pukulan tersebut memiliki durasi 1 ketuk, namun jika jarak hasil perhitungan memiliki nilai setengah dari tempo, maka pukulan tersebut memiliki durasi setengah ketuk, dan seterusnya. Gambar 2 menunjukkan *flowchart* proses pencarian pola.

Durasi hasil pencarian proses sebelumnya belum bisa digunakan dalam penyusunan notasi karena setiap durasi setiap *note* dalam notasi harus memilki nilai $2^n$. Oleh karena itu diperlukan pendekatan agar semua durasi bisa memiliki nilai $2^n$. Pendekatan dilakukan dengan memandingkan durasi hasil perhitungan dengan daftar nilai durasi yang mungkin dimiliki oleh suatu *note*. Durasi hasil perbandingan merupakan durasi dalam daftar yang selisihnya terkecil dengan durasi hasil perhitungan. Durasi hasil perbandingan ini kemudian dibawa ke proses pembuatan *note*.

Proses pembuatan *note* adalah proses pembuatan objek berdasarkan data durasi dari hasil perbandingan. Setelah pencarian durasi setiap *note* diperoleh, proses yang selanjutnya dilakukan adalah penyusunan notasi dalam birama. Proses penyusunan ini dilakukan dengan memasukkan satu per satu *note* ke dalam birama. Karena satu birama hanya boleh terdiri dari nilai tertentu (dalam hal ini nilai = 1), maka setiap ada *note* baru, dilakukan pengecekan apakah

*note* tersebut cukup masuk ke dalam birama yang sudah ada. Jika jumlah nilai dalam birama ditambah dengan *note* baru masih kurang dari 1, maka *note* baru dimasukkan ke dalam birama, namun jika nilai dalam birama ditambah dengan *note* baru sama dengan 1, maka *note* tetap dimasukkan ke dalam birama namun diikuti dengan penambahan birama baru karena birama yang sekarang nilainya sudah genap 1.



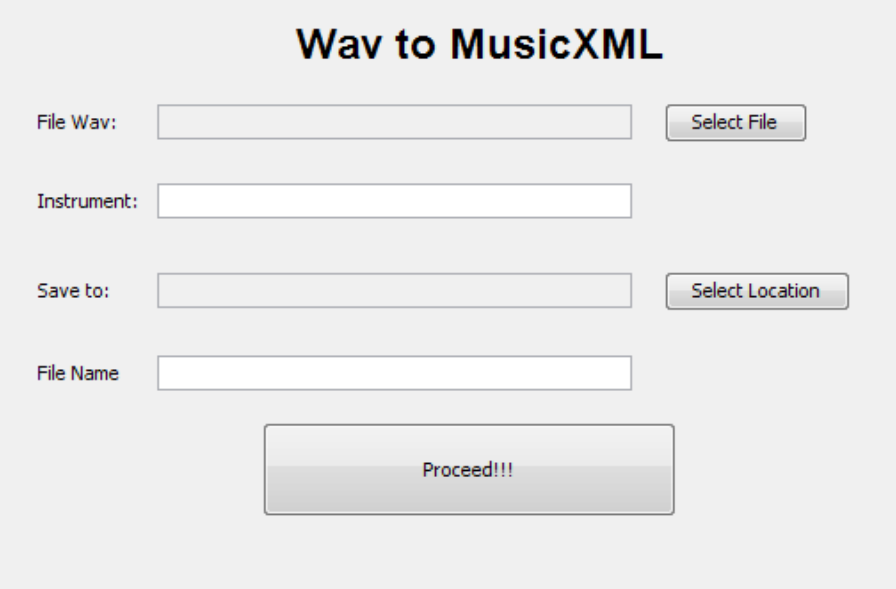

Gambar 3 *Flowchart* proses penyusunan *note* ke dalam birama

Lain halnya jika nilai dalam birama ditambah dengan *note* baru lebih dari 1, maka *note* baru harus dipecah menjadi 2 bagian. Pecahan bagian 1 memiliki nilai sama dengan sisa nilai yang dimiliki oleh birama, dan pecahan 2 memiliki nilai sisanya. Pecahan *note* bagian 1 kemudian diletakkan ke dalam birama sehingga sekarang nilai birama sudah tepat 1. Karena birama sudah tidak dapat diisi *note* baru, maka dibuatlah birama baru. Setelah itu, *note* pecahan bagian 2 diset agar bertipe *rest* atau tidak bunyi dan diletakkan ke dalam birama yang baru. Proses ini diteruskan hingga semua *note* yang telah dibuat dalam proses sebelumnya tersusun rapi di dalam birama-birama. Gambar 3 menunjukkan *flowchart* proses penyusunan *note* ke dalam birama.

Setelah semua note disusun dalam birama-birama, proses yang selanjutnya dilakukan adalah Pembuatan musicXML file berdasarkan data-data tersebut. Proses pembuatan ini dilakukan dengan membuat objek-objek yang terkait dengan atribut-atribut dalam musicXML *file*. Setelah semua atribut diisi, maka konversi dilakukan oleh *library* yang telah disediakan oleh java untuk mentransformasikan objek ke dalam *file* XML.

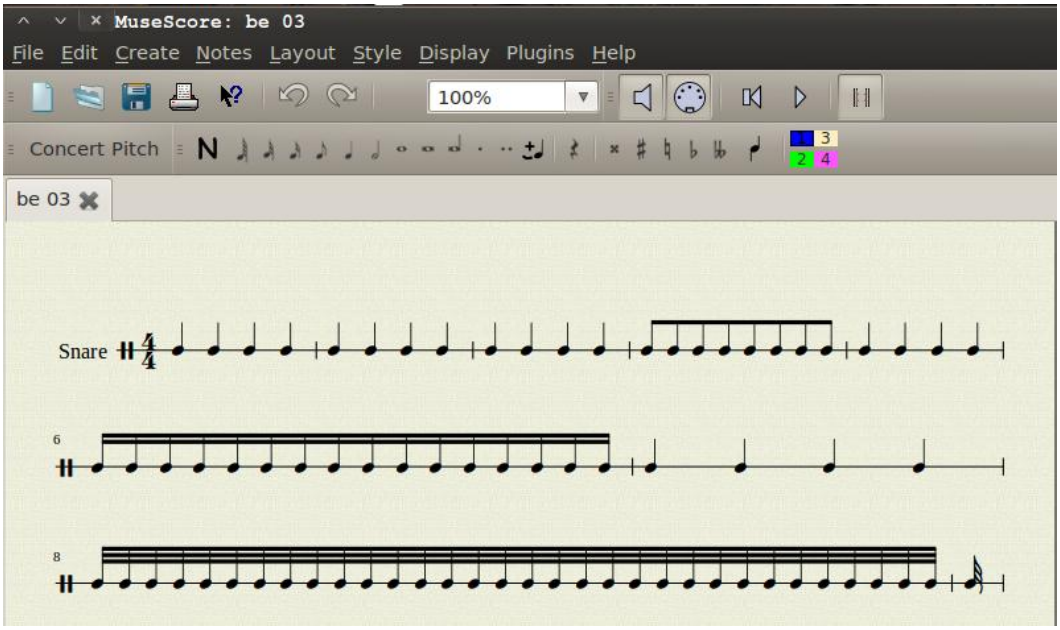
## 3. HASIL DAN PEMBAHASAN

Gambar 4 merupakan tampilan program ketika dijalankan. Tombol "Select File" mengaktifkan dialog untuk memilih file yang akan diproses. Isian baris kedua digunakan untuk pengguna memasukkan nama alat music yang dimainkan. Tombol "Select Location" mengaktifkan dialog untuk memilih lokasi penyimpanan file musicXML hasil keluaran sistem, dan isian baris terakhir merupakan nama file musicXML yang akan dihasilkan sistem. Setelah semua input diisi pengguna, tombol "Proceed!!!" ditekan untuk memulai pemrosesan file rekaman menjadi notasi musik.




Gambar 4 Tampilan awal program

Hasil keluaran sistem yang dibuka menggunakan MuseScore ditunjukkan oleh Gambar. Dari gambar tersebut dapat dilihat bahwa untuk alat musik tidak bernada, garis paranada yang ditampilkan hanya 1 buah dengan *key signature neutral* atau *percussion*, dan nama alat musik *snare*.




Gambar 5 Hasil keluaran sistem

Pengujian terhadap sistem yang telah dibangun dilakukan dengan meminta pemain musik untuk memainkan 6 lagu dengan menggunakan alat musik *snare drum*. Permainan didasarkan kepada notasi musik yang sudah ada sebelumnya. Hasil permainan kemudian dimasukkan ke dalam sistem untuk diperoleh keluarannya. Keluaran sistem tersebut kemudian dibandingkan dengan notasi musik yang digunakan sebagai patokan bermain. Tabel 1 menunjukkan hasil pengujian terhadap sistem. Terdapat 4 poin yang dijadikan penilaian pada pengujian ini antara lain:

1. Total *note*, yaitu banyaknya *note* yang seharusnya terdeteksi. Data ini diperoleh dari menghitung banyaknya *note* yang dimiliki oleh notasi musik awal.
2. *Note* benar, yaitu banyaknya *note* hasil keluaran sistem yang memiliki nilai durasi yang benar. Nilai ini diperoleh dari menghitung banyaknya *note* pada notasi hasil keluaran yang durasinya sama dengan notasi musik awal.
3. Akurasi notasi, merupakan prosentase kebenaran notasi musik. Akurasi ini dihitung membandingkan *note* benar dari hasil keluaran sistem dengan total *note* dari notasi musik awal.

Tabel 1 Hasil pengujian sistem

| Nama File | Nama Alat Musik | Threshold | Total Note | Note Benar | Akurasi |
|---|---|---|---|---|---|
| Sound clip 03 | Snare Drum | 50 | 77 | 77 | 100.00% |
| Sound clip 13 | Snare Drum | 50 | 59 | 59 | 100.00% |
| Sound clip 14 | Snare Drum | 50 | 57 | 57 | 100.00% |
| Sound clip 16 | Snare Drum | 50 | 57 | 57 | 100.00% |
| Sound clip 18 | Snare Drum | 50 | 132 | 129 | 97.72% |
| Sound clip 20 | Snare Drum | 50 | 167 | 167 | 100.00% |
| Rata-rata Akurasi | | | | | 99.62% |

Dari hasil pengujian di atas, diperoleh hasil bahwa akurasi sistem yang menggunakan konsep pertama lebih tinggi hingga mencapai 99.62%. Hasil yang tinggi tersebut diperoleh karena sistem memiliki kemampuan untuk mentolerir ketidakmampuan manusia dalam memberikan jeda yang tepat sama dengan jeda tempo, baik berupa pukulan yang sedikit lebih awal maupun sedikit terlambat. Kemampuan sistem ini ada karena nilai durasi yang dimiliki oleh setiap note dalam sistem adalah hasil pendekatan,

## 4. KESIMPULAN

Telah berhasil dikembangkan sebuah sistem untuk mengolah data audio berupa rekaman suara permainan musik dan menghasilkan keluaran berupa notasi musik. Sistem yang telah dikembangkan memiliki spesifikasi sebagai berikut

1. Sistem mampu mengolah data audio dengan memanfaatkan threshold yang diberikan oleh user.
2. Sistem mampu menghasilkan notasi musik dengan format musicXML, sehingga dapat dibuka dan diolah kembali menggunakan *music editor* yang memiliki kemampuan mengolah *file* dengan format musicXML.
3. Sistem mampu memberikan keluaran dengan akurasi mencapai 99.62%.

## 5. SARAN

Untuk perkembangan sistem sejenis selanjutnya, terdapat beberapa saran yang dapat dijadikan pertimbangan, antara lain:

1. Data audio yang digunakan tidak hanya data dengan format wav.
2. Memiliki kemampuan untuk memisahkan bunyi dengan *noise* untuk mengatasi jika permainan musik diambil dalam suasana yang tidak tenang.
3. Mampu mengenali 2 alat musik atau lebih untuk menciptakan notasi yang lebih kompleks.
4. Dapat mengolah lagu yang memiliki *time signature* selain 4/4.

## UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Sekolah Vokasi Universitas Gadjah Mada dan segenap pemain perkusi Marching Band UGM yang telah memberi dukungan terhadap penelitian ini.

## DAFTAR PUSTAKA

[1] Harnum, J., 2001, *Basic Music Theory: How to Read, Write, and Understand Written Music*, SolUt Press, http://www.allabouttrumpet.com/pdf/BMT/BMT-2nd-ed.pdf, diakses tanggal 22 Desember 2011.

[2] Good, M., 2006, "Lesson for the Adaption of MusicXML as an Interchange Standard", *Proceedings of XML*, Boston.

[3] Klapuri, A., Eorenen, A,, Seppanen, J., Virtanen, T., 2001, "Automatic Transcription of Music", *Symposium on Stochastic Modeling of Music, 14th Meeting of the FWO Reseach Society on Foundation of Music Research,* Belgium, http://www.cs.tut.fi/sgn/arg/klap/ATOM_article_2001.pdf, diakses tanggal 9 November 2011.

[4] Nickol, P., 2004, *Panduan Praktis Membaca Notasi Musik,* PT Gramedia Pustaka Utama, Jakarta.

[5] Foster, S., Schloss, W. A., and Rockmore, A. J., 1982, "Toward an Intelligence Editor of Digital Audio: Signal Processing Methods". *Computer Music Journal*, Vol. 6, No 1, pp. 44-51.

[6] McNab, R. J., Smith, L. A., and Witten, I. H., 1996, "Signal Processing for Melody Transcription". *Proc. 19th Australasian Computer Science Conf.,* pp. 301-307, Melbourne.

[7] Correa, J. P. B., 2003, "Towards the Automated Analysis of Simple Polyphonic Music: A Knowledge-based Approach", *PhD Thesis,* Department of Electronic Engineering, Queen Mary, University Of London, London.

[8] Collins, N., 2005, "A Change Discrimination Onset Detector with Peak Scoring Peak Picker and Time Domain Correction", *6th International Conference on Music Information Retrieval,* University of London, London, http://www.music-ir.org/evaluation/mirex-results/articles/onset/collins.pdf, diakses tanggal 10 November 2011.

[9] Krishnamurthy, K., 2008, "Generation of Control Signals Using Pitch and Onset Detection for an Unprocessed Guitar Signal", https://ccrma.stanford.edu/~kapilkm/220c/final_paper.pdf, diakses tanggal 10 November 2011.

[10] Gainza, M., Lawlor, B., and Coyle, E., 2004, "Onset Detection and Music Transcription for the Irish Tin Whistle", *Proceeding, Irish Signals and Systems Conference, ISSC: 2004,* Queen's Univeristy Belfast, Northen Ireland.

[11] Costantini, G., Todisco, M., Perfetti, R., Basili, R., Casali, D., 2010, "Memory Based Automatic Music Transcription System for Percussive Pitched Instruments", *Proc. Of the 1st International Multi-Conference on Complexity, Informatics, and Cybernetis,* Florida.